REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Service, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC 20503.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
AUGUST 2010	Conference Paper	May 2008 – March 2010

4. TITLE AND SUBTITLE

CONGESTION CONTROL AND FAIRNESS IN WIRELESS SENSOR NETWORKS

5a. CONTRACT NUMBER
In House

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62702F

6. AUTHOR(S)

Swastik Brahma, Mainak Chatterjee, and Kevin Kwiat

5d. PROJECT NUMBER
23G4

5e. TASK NUMBER
IH

5f. WORK UNIT NUMBER
01

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

AFRL/RIGG
525 Brooks Road
Rome NY 13441-4505

University of Central Florida
Department of Electrical Engineering and Computer Science
Orlando, FL32816

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RIGG
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TP-2010-29

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved For Public Release; Distribution Unlimited. PA #: 88ABW-2009-4435
Date Cleared: 22-Oct-2009

13. SUPPLEMENTARY NOTES
© 2010 IEEE. This paper was published in the Proceedings of the 2010 IEEE International Conference Pervasive Computing and Communications. This work is copyrighted. One or more of the authors is a U.S. Government employee working within the scope of their Government job; therefore, the U.S. Government is joint owner of the work and has the right to copy, distribute, and use the work. All other rights are reserved by the copyright owner.

14. ABSTRACT
A distributed congestion control algorithm for tree based communications in wireless sensor networks is proposed that seeks to adaptively assign a fair and efficient transmission rate to each node. In the algorithm described, each node monitors its aggregate output and input traffic rates. Based on the difference of the two, a node then decides either to increase or decrease the bandwidth allocable to a flow originating from itself and to those being routed through it. Since the application requirements in sensor network follow no common trait, the proposed design abstracts the notion of fairness, allowing for the development of a generic utility controlling module. Such separation of the utility and fairness controlling modules enables each one to use a separate control law, thereby portraying a more flexible design.

15. SUBJECT TERMS
Wireless Networks, Game Theory, Attack Detection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Kevin A. Kwiat
U	U	U	UU	7	19b. TELEPHONE NUMBER (*Include area code*) N/A

POSTPRINT

Congestion Control and Fairness in Wireless Sensor Networks

Swastik Brahma and Mainak Chatterjee
Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816
{sbrahma, mainak}@eecs.ucf.edu

Kevin Kwiat
Air Force Research Laboratory
Information Directorate
Rome, NY
kevin.kwiat@rl.af.mil

***Abstract*—In this paper we propose a distributed congestion control algorithm for tree based communications in wireless sensor networks, that seeks to adaptively assign a *fair* and *efficient* transmission rate to each node. In our algorithm, each node monitors its aggregate output and input traffic rates. Based on the difference of the two, a node then decides either to increase or decrease the bandwidth allocable to a flow originating from itself and to those being routed through it. Since the application requirements in sensor network follows no common trait, our design *abstracts* the notion of fairness, allowing for the development of a generic utility controlling module. Such separation of the utility and fairness controlling modules enables each one to use a separate control law, thereby portraying a more flexible design. The working of our congestion control is independent of the underlying routing algorithm and is designed to adapt to changes in the underlying routing topology. We evaluate the performance of the algorithm via extensive simulations using an event-driven packet level simulator. The results suggest that the proposed protocol acquires a significantly high goodput of around 95% of the actual transmission rate, converges quickly to the optimal rate, and attains the desired fairness.**

I. Introduction

Transmission control protocols are a key enabling technology in many of today's sensor network applications. Applications like those of habitat monitoring [9], structural health monitoring [5], image sensing [7] are high data-rate applications that heavily rely on congestion control techniques which are an integral part of any transmission control protocol. Well designed congestion control techniques allow efficient transmission of significant volumes of data from a large number of nodes along one or more routes towards the data processing centers (usually known as 'sink'). In these high data-rate applications, often bulk data is generated in addition to the constantly sensed data. For example, in structural health monitoring, each sensor measures structural vibration continuously at a certain rate. When the sensors detect a significant anomaly, they generate and send out date at a much higher rate. Without congestion control, under such traffic characteristics, congestion collapse is inevitable.

Furthermore, providing fairness among flows is also highly desirable. However, the notion of fairness is hard to ground in sensor networks since the application requirements follow no common trait. How exactly available bandwidth will be apportioned among the flows is dictated by the requirements of the application. Thus, it would be beneficial if the notion of fairness can be *abstracted* while designing a congestion control scheme for sensor networks. This would facilitate *de-coupling* the control of *utilization of the network* from the management of achieving *fairness among different flows*. Such a separation would allow easy implementation of different fairness models according to the needs of the application at hand without considering the design of the module controlling utilization of the network. Moreover, such a design would also be beneficial for sensor networks which can support multiple concurrent applications and also systems with multiple users [6], where there is a need to treat flows from different applications or users in an inequitable manner. In general, de-coupling of the two modules simplifies the design and analysis of each one by reducing the requirements imposed, and enables re-designing one of the modules without considering the other. Though, such separation of the utility and the fairness controller has been proposed for traditional wired networks in [4], designing such a scheme in the regime of sensor networks is not a trivial extension of previous work.

In this paper, we propose a distributed and adaptive mechanism for controlling congestion in sensor networks which seeks to find an optimal transmission rate for the nodes, that is both fair and maximally efficient. In our scheme, we use two separate modules to control *utility* of the network and *fairness* among flows. Each node monitors its aggregate output rate and the aggregate input rate. Based on the discrepancy between the two, a node first computes the aggregate increase (if the output rate is more) or decrease (if the input rate is more). Then the fairness controlling module acts on this aggregate change required and apportions the same into individual flows to achieve the desired fairness. The key advantages of our proposed algorithm can be summarized as follows.

- The proposed protocol adjusts its aggressiveness according to the spare bandwidth in the network. This reduces oscillations, provides stability, and ensures efficient utilization of network resources.
- Decoupling the utility and fairness controlling module in our protocol opens new avenues for service differentiation using schemes that provide desired bandwidth sharing.
- The scheme also supports multiple concurrent applications and is also highly robust to changes in the under-

This research was sponsored by the Air Force Office of Scientific Research (AFOSR) under the federal grant No. FA9550-07-1-0023. Approved for Public Release; distribution unlimited: 88ABW-2009-4435 22 Oct 09.

978-1-4244-5328-3/10/$26.00 ©2010 IEEE

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.

lying topology and routing dynamics and can adapt to changes in the same.
- The protocol has an additional advantage of improving channel quality by inducing a *phase-shifting effect* among neighboring nodes, which introduces a slight jitter to the periodicity of the application thereby breaking the synchronization among periodic streams of traffic.

The rest of the paper is organized as follows. Section II provides a brief literature review of related works in this area. The congestion control algorithm is presented in section III. Section IV analyzes the steady state performance of the algorithm via extensive simulations. Section V concludes the paper.

II. Related Research

Prior work in sensor networks literature has broadly looked at two qualitatively different problems, viz, *congestion mitigation* and *congestion control*. Congestion mitigation deals with regulating the transmission rates of the nodes when the aggregate traffic exceeds the network capacity so that the network goodput and fairness degrade gracefully. This is different from congestion control, which seeks to find an optimal fair rate for the sensor nodes that is also maximally efficient. In this case, when the nodes transmit data at the optimal rate, the network is fully utilized, and the per node goodput is close to the sending rate.

Adaptive Rate Control (ARC) [11] monitors the injection of packets into the traffic stream as well as route-through traffic. Each node estimates the number of upstream nodes and the bandwidth is split proportionally between route-through and locally generated traffic, with preference given to the former. The resulting bandwidth allocated to each node is thus approximately fair. Also, reduction in transmission rate of route-through traffic has a backpressure effect on upstream nodes, which in turn can reduce their transmission rates. In [6], the authors propose the Rate Controlled Reliable Transport protocol (RCRT). This protocol is built for loss-intolerant applications that require *reliable* transport of data from the source nodes to the sink. RCRT uses end-to-end explicit loss recovery by implementing a NACK based scheme. Furthermore, RCRT places all congestion detection and rate adaptation functionality in the sinks, thereby producing a centralized congestion control scheme. In [10], the authors propose Congestion Detection and Avoidance (CODA). CODA uses several mechanisms to alleviate congestion. In *open-loop hop-by-hop backpressure*, when a node experiences congestion, it broadcasts back-pressure messages upstream towards the source nodes, informing them of the need to reduce their sending rates. In *closed-loop multi-source regulation*, the sink asserts congestion control over multiple sources and uses acknowledgements to determine their sending rates when traffic load exceeds the channel capacity. Fusion [3] is another congestion mitigation technique that uses queue lengths to detect congestion. Fusion uses three different techniques to alleviate congestion, viz, hop-by-hop flow control, rate limiting, and a prioritized MAC. In [8], the authors proposed the Interference Aware Fair Rate Control protocol (IFRC) which is a distributed rate allocation scheme that uses queue sizes to detect congestion, and further shares the congestion state through overhearing. Congestion control and fairness for many-to-one routing in sensor networks [2] is another rate allocation scheme that uses different mechanism than IFRC. Both IFRC and [2] are tangentially related to our work in the sense that they attempt to find a optimal transmission rate for all the nodes that avoid congestion collapse. Note that, our algorithm has greater flexibility than IFRC and [2], since many different traffic allocation policies can be implemented in our congestion control scheme, without changing the basic congestion control module (the utility controller). Moreover, IFRC suffers from the additional drawback of having sophisticated parameter tuning for stability, unlike ours.

III. Congestion Control Design

Let us consider a set of N sensor nodes, numbered 1 through N. In the simplest version of the problem, each node has an infinite amount of data to be sent to a single base station. The nodes can originate data traffic, as well as route traffic originated by other nodes. Thus each node can act both as a *source*, and a *router*. The nodes sample the environment at periodic intervals, encodes the information into data packets, and sends them out to a central *base station* or *sink*. Let the flow originating from Node i be f_i, and let r_i be the rate at which flow f_i is injected into the network. We seek to adaptively assign flow f_i a *fair* and *efficient* rate r_i. Note that, r_i is the rate at which node i injects flow f_i into the network, and does not include the rate at which node i forwards traffic.

We assume that the sensor nodes run a contention based MAC protocol. The default MAC in TinyOS, a widely used sensor network operating system, uses carrier sense for collision avoidance. Our implementation is based on this MAC, though it can be extended to other MAC protocols as well, such as those that use RTS/CTS [13]. We further assume that the MAC layer provides link-layer retransmissions. Our current design works well in a regime where the wireless loss rate over the communication links are such that link layer retransmissions recover from most packet losses.

We further assume that the sensor nodes run a routing protocol [12] that builds a routing tree rooted at the base station (or sink). The working of our congestion control algorithm is not dependent on the exact choice of the routing tree formed. However, the performance of the algorithm would benefit from a tree formed based on link quality metrics. In the rest of the paper, we assume that a routing tree has been formed by the routing protocol rooted at the sink. Our congestion control algorithm is able to adapt to changes in the underlying routing topology.

We now develop a simple distributed algorithm for congestion control in sensor networks. Before we begin we first define a few terms.

Feedback Delay: Feedback delay of a flow is the time interval, measured starting from the time node i starts transmitting flow f_i at a rate r_a to the time the control signal arrives and changes

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.

the rate to r_b.

Gateway Node: A node is called a gateway node if it is one hop away from the sink.

Control Interval: Control interval is the time period over which a node takes a control decision regarding the increase or decrease in the transmission rates of the flows originated by itself and of the flows being routed through it. In our implementation this interval has been taken to be the average feedback delay of the flows passing through the gateway node.

A. Congestion Control Algorithm

The congestion control algorithm can be explained by the following steps executed at each node every control interval:

1) Measure the average rate r_{out} at which packets can be sent from the node, the average aggregate input rate r_{in}, and Q, the minimum number of packets in the output queue seen by an arriving packet in a control interval.
2) Based on the difference between r_{out} and r_{in}, and Q, compute Δr, which is the total change in aggregate traffic required as: $\Delta r = \alpha \times (r_{out} - r_{in}) - \beta \times (\frac{Q}{t_{CI}})$ where, α, β are constants, and t_{CI}
3) Apportion Δr into individual flows to achieve fairness.
4) Compare the bandwidth computed for each flow with the bandwidths advertised by its parent. Use and propagate the smaller rate upstream.

An important point that needs to be clarified is when does the congestion control procedure get invoked at the various nodes. The congestion control algorithm gets invoked every control interval at the gateway nodes. For any other node, the algorithm is invoked when the transmission rate of its parent changes. This essentially makes the congestion control to be invoked at *all* nodes every control interval. We will elaborate further on the rationale behind such a control behavior and on the estimation of the control interval later in Section III-A6.

The congestion control algorithm outlined above requires:

1) estimation of average aggregate output rate.
2) estimation of average aggregate input rate.
3) computation of the total change in aggregate traffic required (Δr) to control efficiency.
4) apportioning Δr into individual flows to obtain desired fairness.
5) propagation of rate upstream.
6) estimation of the control interval.

Let us now discuss the working of each one of the above mentioned requirements.

1) Estimation of Average Output Rate: Let t_{out} be the time required to transmit a packet, measured starting from the time the packet was sent by the network layer to the MAC layer to the time when the MAC layer notifies the network layer that the packet was successfully transmitted. This is shown in Figure 1, where we assume that the MAC protocol in use is CSMA/CA with an acknowledgement based scheme. Then, we note that the effective rate r_{out} packets per second is the inverse of the time interval t_{out} seconds, i.e, $r_{out} = \frac{1}{t_{out}}$.

The value of t_{out} obtained per packet transmitted is one particular instance of the average time taken to transmit a packet. We compute the average value by using the exponential moving average:

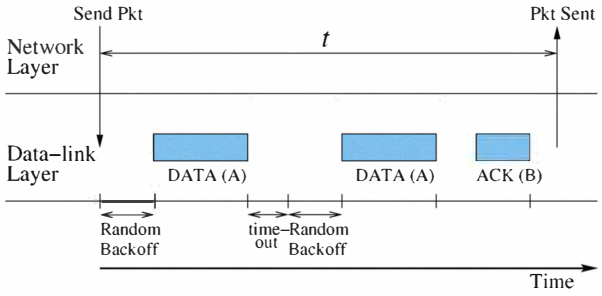

Fig. 1. Determination of time t taken to successfully transmit a data packet, considering CSMA. The sender of a packet is denoted in parenthesis.

$$\overline{t_o^i} \qquad \overline{\qquad} \tag{1}$$

where, $\overline{t_o^i}$
variable t_{ou} in the i^{th} iteration, α_{out} is the weight and T_{out} is the current value of the variable t_{out} Calculating $r_{out} = \frac{1}{t_{out}}$ gives us the average rate at which packets can be transmitted from a particular node.

2) Estimation of Average Input Rate: Let t_{in} the inter packet arrival time at a node, measured starting from the time a packet was enqueued to the time when the next packet is successfully enqueued. Then, the effective aggregate input rate r_{in} at a node is the inverse of the time interval t_{in}, i.e, $r_{in} = \frac{1}{t_{in}}$.

We compute the average value of t_{in} moving average:

$$\overline{t_{in}} - \quad \alpha_{in} \quad \cdot T_{in} + (1 - \alpha_{in}) \cdot \overline{t_{in}} \tag{2}$$

where, $\overline{t_i^i}$
variable t_{in}
the current value of the variable t_{in}. $\frac{1}{t_{in}}$
gives us the average aggregate input rate at a node.

3) Controlling Efficiency: In controlling efficiency, we seek to maximize link utilization, while minimizing buffer drop rates and persistent queues. The efficiency controlling component considers only the aggregate traffic and does not take into account fairness issues.

The efficiency controller computes the required increase or decrease of the aggregate transmission rate of the traffic (Δr) in a control interval (in packets/second). This is computed as:

$$\Delta r = \alpha \times (r_{out} - r_{in}) - \beta \times (\frac{Q}{t_{CI}}) \tag{3}$$

where, t_{CI}
constant parameters and Q is the persistent queue size. We use 0.4 and 0.226 as values of α and β respectively, based on the work done in [4]. Q is computed as the minimum number of packets seen by an arriving packet in a control interval. Note that the quantity, $(r_{out} - r_{in})$, can be positive, negative or equal to zero. When $(r_{out} - r_{in}) > 0$, the link is underutilized and positive feedback needs to be send to increase the transmission rates of the flows. When $(r_{out} - r_{in}) < 0$, link is congested and negative feedback is required to decrease the transmission rates. For the case when $(r_{out} - r_{in})$ is equal to zero, i.e,

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.

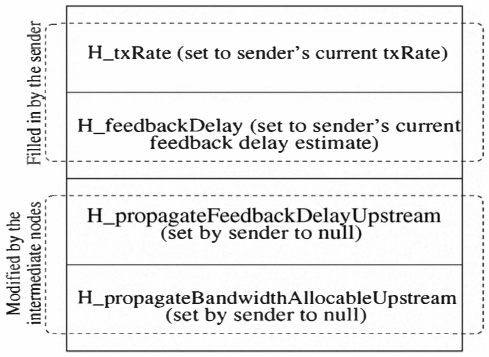

Fig. 2. Congestion Header

the input capacity matches the link capacity, we have to provide feedback in a manner which drains the persistent queue size Q. This is why the aggregate feedback has been made proportional to Q.

4) Controlling Fairness: The task of the fairness controlling component is to apportion the feedback computed by the efficiency controlling component into the individual flows. For sake of simplicity in this section we treat all flows equally. Section III-B discusses further on differentiable treatment of flows by the fairness controller. Though there could be other choices, we use AIMD to make the system converge to fairness. There are three cases:

- $\Delta r > 0$**:** In this case the positive feedback to be sent is divided equally among the flows so that increase in throughput of all the flows is the same. If we assume that there are n flows comprising the route-through traffic of a node, then each one of the flows should get $1/(n+1)^{th}$ fraction of the spare bandwidth, considering that the node also originates a traffic of its own. Thus we can write:

$$r_i(t+1) = r_i(t) + \frac{\Delta r}{n+1} \tag{4}$$

- $\Delta r < 0$: In this case the negative feedback to be sent is allocated in such a way that the decrease in throughput of a flow is proportional to its current throughput. Thus,

$$r_i(t+1) = m \cdot r_i(t) \tag{5}$$

  where, $0 < m < 1$. Here m is inversely proportional to the magnitude of Δr, $|\Delta r|$. This means that, as the discrepancy between the output rate and the input rate increases, the throughput of the flows is cut down more and more aggressively.
- $\Delta r = 0$: This case corresponds to the efficiency being around optimal. Bandwidth shuffling is done in this case, such that, the total traffic rate, and consequently the efficiency, does not change, yet the throughput of each individual flow changes gradually to approach the flow's fair share.

5) Propagation of Rate Upstream: As mentioned earlier, congestion controlled is invoked at the *gateway nodes* every control interval. For all the other nodes congestion control is invoked when the transmission rate of the parent node changes. It is quite evident that, this, in effect, leads to the congestion control being invoked at all the nodes every control interval. In order to propagate the congestion signal upstream, starting from the gateway nodes, we make use of the broadcast nature of the wireless medium, which enables a child node to overhear transmissions of its parent. When a node (say j) transmits a packet of a flow f_i, it writes into a header (H_propagateBandwidthAllocableUpstream as shown in Figure 2) in the packet, the minimum of the bandwidths allocable to the flow by itself (which it computes when its congestion control is invoked) and that advertised by its parent. Note that, the bandwidth advertised by its parent is, in turn, the minimum of the bandwidth allocable to f_i by the nodes downstream to node j. Thus, the transmission rate for flow f_i that node j writes into the header of a packet of f_i, is the minimum bandwidth allocable to the flow among nodes downstream starting from node j. In this fashion, the feasible transmission rate, that ultimately reaches node i (which originates flow f_i), is the minimum bandwidth allocable to the flow by the intermediate nodes along the path of the flow.

6) Estimation of Control Interval: Control theory states that a controller must react as quickly as the dynamics of the controlled signal; else, the controller will lag behind the system being controlled. Thus, the controller of our congestion control scheme makes a single control decision every average *feedback delay* period (averaged over the feedback delays of the flows passing through a node). This is motivated by the need to observe the results of the previous control decisions before attempting a new control. For example, if an intermediate node tells an upstream node to increase its transmission rate, the former should wait to see how much spare bandwidth remains before telling it to increase again.

As per the definition of *feedback delay*, it is equal to delay experienced by the packets of the flow to go from the source node to one of the gateway nodes plus the time taken by the congestion signal to propagate upstream from the gateway node back to the source node. Now, the time taken by the packets to travel from source nodes to the gateway nodes is comprised of the queueing delays of the packets plus the average time taken to transmit a packet successfully by the MAC layer at the various intermediate hops. Further, time taken by the congestion signal to propagate upstream from the gateway node to the source node is comprised of the summation of the inter packet arrival interval of the packets of the flow at the head of the output queue at the intermediate nodes and the time taken to transmit a packet successfully by the MAC layer at these nodes. To understand why, recall that the congestion signal propagates upstream by utilizing the broadcast nature of the wireless medium, enabling a child node to overhear transmission of its parent. Thus, when the minimum bandwidth allocable to a flow changes at a parent node, its child comes to know about it *at most* after the instantaneous inter packet arrival interval of the packets of the flow at the head of the output queue at the parent plus the time taken to transmit a packet by the MAC layer at the same.

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.

Now, what is required is to compute the above mentioned total delay in a distributed manner. For this, each node keeps a moving average of the queueing delay, inter packet arrival interval, and twice the MAC packet transmission time for each flow passing thorough it. When a node (say j) transmits a packet of a flow k, it writes into a header (H_propagateFeedbackDelayUpstream as shown in Figure (2) in the packet the summation of the three terms as estimated by itself for flow k and the value for feedback delay for flow k as advertised by its parent. The value for feedback delay advertised by its parent is in turn the summation of the three terms of the nodes downstream to j. In this fashion, the source node can be made aware of the feedback delay of the flow it is originating. The source node, on its part, maintains the feedback delay experienced by its flow and sends them out to the intermediate nodes via a header (H_feedbackDelay as shown in Figure (2) in every packet.

As mentioned earlier, gateway nodes are those which are one hop away from the sink and traffic from every other node has to go via one of these nodes to get to the sink. The gateway nodes uses the average feedback delay of the flows passing through them as the control interval. For the other nodes in the network, the congestion control procedure is invoked when the transmission rate of its parent changes. In this scheme, effectively, the congestion control algorithm gets invoked for all the nodes every average feedback delay of flows passing through the gateway node, which is the control interval.

B. Discussion: The Phase Shifting Effect

From a control theory perspective, the system forms a closed loop and is in constant oscillation. When the aggregate traffic input rate at a node becomes more than its aggregate output rate, the node decreases the bandwidth allocable to all its upstream nodes, including itself. This gets propagated through the network, and ultimately the queues start draining. This causes less nodes to transmit, thus reducing interference among neighboring nodes and the time taken to successfully transmit a packet decreases. The node, then advertises increase in bandwidth of the flows, and gradually the transmission rate of the flows increase, thus luring congestion. This cycle goes on repeating and the instantaneous transmission rates of the nodes fluctuates around the optimal value. This fluctuation has an important effect of *shifting phase* among neighboring nodes. Thus, the nodes will be injecting packets at different times. This helps reduce interference among neighboring nodes, thus, improving channel quality and the average retransmission count of the nodes at the MAC layer accordingly decreases.

IV. Performance Evaluation

To evaluate the performance of the congestion control algorithm, we implemented a packet level simulator in Java. The simulator is event-driven and implements CSMA/CA as the MAC protocol.

Table I shows the parameter values used in the experiments. The weights used in the exponential moving average calculation of the output and input rate (see Section III-A1 and Section III-A2) is set to 0.1.

TABLE I
SIMULATION PARAMETER VALUES USED IN THE EXPERIMENTS

Parameter	Values
Channel Bit-rate	38.4 kbps
Topology considered	10×10 grid
Link loss probability	0.1
Initial pkt generation rate	1 pkts/sec
Maximum Retx threshold	7
Queue Size	25
Data pkt size	64 bytes
Beacon pkt size	10 bytes
MAC ACK pkt size	5 bytes

A. Goodput and Fairness

Figure 3 shows the per-flow goodput received at the sink. Each bar represents the average rate of transmitted packets from the corresponding node. The darker section of each bar represents the average rate at which packets from that node were received at the base station. Packet losses account for the rest (the remaining lighter section of the bar). We make an important observation from this graph. This figure validates the basic design of the congestion control algorithm, and shows that nodes receive approximately fair rates and fair goodput.

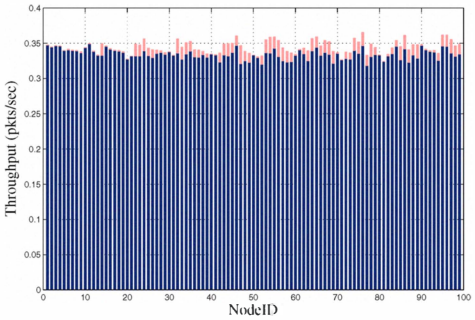

Fig. 3. Per flow goodput in the 100 node experiment

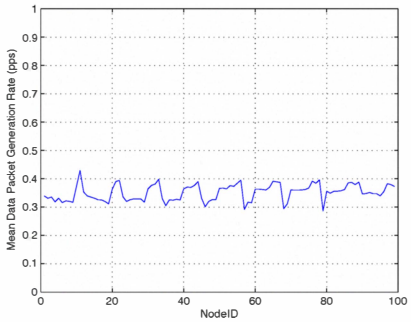

Fig. 4. Mean Data Packet Generation Rates per node

B. Data Generation Rate Oscillations

One of the design criterions of the congestion control algorithm is to receive equitable amounts of data from all the nodes at the sink, which in other words requires an average equal transmission rates for the nodes. Fig 4 tends to bring

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.

out this fact, wherein we observe that the long term average transmission rates for all the nodes falls within a narrow range of values. The spikes that we observe in Figure 4 are mainly at nodes along the diagonal of the simulation grid. This is due to the fact that traffic from relatively lesser number of nodes follow this route. Since the bandwidth along a route is split up equally among competing flows, hence these nodes tend to achieve a higher average transmission rate than the other nodes in the network.

Fig 5 shows the variation of data packet generation rate at node 2 with time. From the graph we can see that the congestion control scheme is able to adapt and obtain the effective transmission rate quickly. As can be seen from the graph, the transmission rate of the node oscillates around a mean. This oscillation around the mean causes phase shifting among neighboring nodes, as explained in Section III-B, thus reducing interference and improving channel quality.

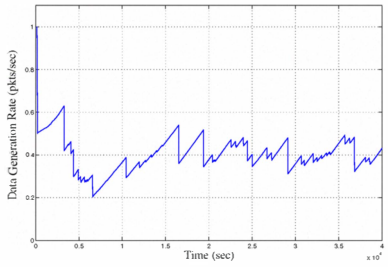

Fig. 5. Data Packet Generation Rate fluctuation (Node 2)

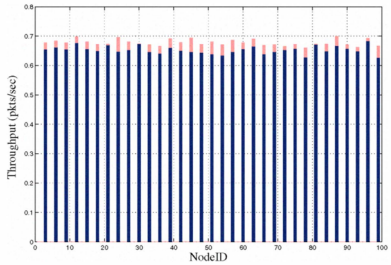

Fig. 6. Per flow goodput with only a subset of senders. Nodes whose ID's are multiples of 3 were only allowed to transmit data.

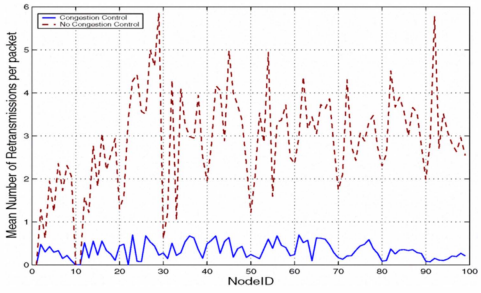

Fig. 7. Mean retransmissions per node due to interference

C. Subset of Senders

Figure 6 shows the per flow goodput with a subset of senders. Those nodes whose ID's are multiples of 3 were only allowed to transmit data. As can been seen from the graph, all nodes receive a fair rate. Note that the per flow goodput is significantly higher than when all nodes transmit. This is because the protocol adapts to the increased overall available capacity and allocates it fairly to the transmitting nodes.

D. Link Layer Retransmissions

Figure 7 shows that without congestion control each packet is retransmitted due to interference a maximum of about 6 times, a number dependent on many factor such as the topology, data packet generation rate and size of the network. In general, number of retransmissions, and therefore losses, increases with the depth of the network. The graph where congestion control is implemented shows a small number of retransmission per packet and this number roughly stays constant for all nodes in the network.

V. Conclusions

The paper presents a distributed algorithm for congestion control in wireless sensor networks that seeks to assign a fair and efficient rate to each node. The algorithm requires each node to monitor their aggregate input and output traffic rate, based on the difference of which they decide to increase or decrease the transmission rates of itself and its upstream nodes. The utilization controlling module computes the total increase or decrease in traffic rate. The fairness module decides on how exactly to apportion the total change in traffic rate required among the flows. The utilization controller is indifferent to it, thus abstracting the notion of fairness, and allowing the fairness module to assume differential bandwidth allocation policies. We tested our algorithm on an event-driven packet level simulator. The results indicate that the congestion control can achieve remarkably high goodput, is able to attain fairness for all nodes in the network, and can acquire the optimal transmission rate quickly.

References

[1] D. Chiu, R. Jain, *Analysis of the increase and decrease algorithms for congestion avoidance in computer networks*, Computer Neworks and ISDN Systems 17, page 1-14, 1989.

[2] C.T. Ee, R. Bajcsy, *Congestion Control and Fairness for Many-to-One Routing in Sensor Networks*, In Sensys 2004.

[3] B. Hull, K. Jamieson, H. Balakrishnan, *Mitigating Congestion in Wireless Sensor Networks*, In Sensys 2004.

[4] D. Katabi, M. Handley, C. Rohrs, *Congestion Control for High Bandwidth-Delay Product Networks*, In Sigcomm 2002.

[5] J. Paek, K. Chintalapudi, J. Cafferey, R. Govindan, S. Masri, *A Wireless Sensor Network for Structural Health Monitoring: Performance and experience*, In EmNetS 2005.

[6] J. Paek, R. Govindan, *RCRT: Rate-Controlled Reliable Transport for Wireless Sensor Networks*, In Sensys 2007.

[7] M. Rahimi, R. Baer, J. Warrior, D. Estrin, M. B. Srivastava. *Cyclops: In situ Image Sensing and Interpretation in Wireless Sensor Networks* In SenSys 2005.

[8] S. Rangwala, R. Gummadi, R. Govindan, K. Psounis, *Interference-Aware Fair Rate Control in Wireless Sensor Networks*, In proceedings of ACM SIGCOMM Symposium on Network Architecture and Protocols, 2006.

[9] R. Szewczyk, A. Mainwaring, J. Polastre, D. Culler, *An Analysis of a Large Scale Habitat Monitoring*, In Sensys 2004.

[10] C. Wan, S.B. Eisenman, A.T. Campbell, *CODA: Congestion Detection And Avoidance in Sensor Networks*, First ACM conference on Embedded Networked Sensor Systems, Nov 2003.

[11] A. Woo, D. Culler, *A Transmission Control Scheme for Media Access in Sensor Networks*, Seventh Annual International Conference on Mobile Computing and Networking, pp 221-235, July 2001.

[12] A. Woo, T. Tong, D. Culler, *Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks*. In SenSys 2003.

[13] W. Ye, J. Heidemann, D. Estrin, *An Energy-Efficient MAC Protocol for Wireless Sensor Networks*, In Infocom 2002.

Authorized licensed use limited to: ROME AFB. Downloaded on August 16,2010 at 11:45:13 UTC from IEEE Xplore. Restrictions apply.